UNIVERSIDADE D
COIMBRA

Sistemas Distribuídos

ucBusca meta 1

2019/2020

Ricardo Martins 2017246268

Rodrigo Rafael 2017246279

Arquitetura do sistema

O multicast server é o servidor que mantém registos permanentes da aplicação, podem existir n cópias deste servidor. As comunicações deste servidor são feitas através de conexões UDP multicast, para receber pedidos e enviar respostas, ou por TCP, para atualizar os dados do sistema e, se necessário partilha de carga entre 2 ou mais servidores.

Existem dois servidores RMI (um principal e um backup) que fazem a conexão com os clientes para receber pedidos, transformam os pedidos para obedecer ao protocolo de comunicação estabelecido e enviam os pedidos para o multicast, que trata de os processar e escrever uma resposta no mesmo protocolo e enviar para o RMI server.

Quando o multicast principal deixa de funcionar o backup assume como principal.

Funcionamento Multicast

Cada multicast server tem um objeto storage para guardar todos os dados persistentes, este objeto tem um hash map para tabelar as pesquisas e os URLs e outro para guardar todas os sites que referencia determinado URL.

Existe uma classe user, que guarda todas as notificações não entregues, bem como o histórico de pesquisas, as suas permissões e informações pessoais.

Para lidar com os pedidos existe uma request queue para os armazenar, e existe uma Thread(manageRequests) que processa um pedido de cada vez envia a resposta e vai buscar outro pedido à queue, se esta não estiver vazia. Se a queue não tiver pedidos a thread fica à parada à espera de um “notify” para acordar e continuar a trabalhar.

O web crawler utilizado para a indexação de websites é também uma thread que funciona com uma queue, quando o multicast recebe um pedido de indexação, adiciona o link indicado à fila para este ser indexado.

Cada servidor sabe a qualquer momento que servidores se encontram ativos através dos keepAlives, uma mensagem recebida através do grupo multicast, onde cada servidor envia a sua carga de endereçamento, e as suas configurações TCP. Se um servidor passar mais de 30s sem enviar um keepAlive significa que está inativo. Os keepAlives são enviados em intervalos de 1s.

Cada servidor tem uma thread à espera de ligações TCP, quando recebe uma abre uma thread “connection” que recebe a informação e dá início ao merge da informação recebida (alterações nos hash maps de pesquisa e referencia e urls para indexar), com a sua informação atual. Cada servidor estabelece uma ligação TCP, através da thread ShareInfo, com todos os outros multicast, envia as diferenças em relação à última atualização e fecha o socket. No lado do servidor que recebe as atualizações dá início ao merge para atualizar a informação.

Funcionamento RMI

1) 2 Servidores RMI

Neste projeto existem 2 servidores RMI, um primário e outro secundário. Apenas o primário interage com os clientes RMI, no entanto, quando este falha, existe o secundário para o substituir, não necessitando de iniciar nova sessão com o cliente. Estes servidores não armazenam nada localmente, apenas disponibilizam o conjunto de métodos acessíveis aos clientes, através da interface remota. Quando um método é chamado, o servidor RMI traduz esse pedido num datagrama UDP enviado para os servidores Multicast. Depois disto, o servidor RMI espera pela resposta dos Multicast, caso esta não exista, devido a alguma falha, o servidor RMI volta a enviar o pedido num datagrama UDP. As avarias, dos servidores Multicast, inferiores a 30 segundos não serão detetadas pelos clientes. Todos os métodos são throw java.rmi.RemoteException e para lidar com esta exceção, o cliente liga-se de novo ao RMI.

int addPort() - incrementa o porto

boolean logout() - faz o log out de um cliente online

int register(String username, String password) – regista um novo user

int login(String username, String password) – efetua o login de um user

String historic(String user) – retorna o histórico de pesquisas do user

String pagesList(String url) – retorna a lista de url's onde o url em questão foi mencionado

String tenMostImportant() – retorna a lista dos url's mais encontrados

String tenMostSearched() - retorna a lista das dez pesquisas mais efetuadas por todos os users

boolean givePrivileges(String username, String username1) – permite ao admin atribuir privilégios de admin a outro user

String searchWeb(String keyword, String username) – retorna a lista de resultados de uma pesquisa de acordo com as keyword's que o user introduz

void newClient(int port, String myHost)

void ping()

String newURL(String url) – permite ao admin introduzir um novo url para este ser indexado

String verifyNotification(String user) – retorna a lista de notificações de um user, caso existam.

2) Cliente RMI

Neste cliente, é onde os users podem aceder às funcionalidades do ucBusca. O cliente RMI limita-se a chamar os métodos remotos do servidor RMI. A interface do cliente RMI, ao invés da interface do servidor, é bastante simples, apenas contém dois métodos: void notification(String message); String getUser().

Protocolo de comunicação RMI-Multicast

Os pedidos enviados e os valores das respostas não podem conter " | ", " ; " nem "\n" visto que se tratam dos separadores escolhidos.
; -> serve para separação dos conjuntos chave-valor
| -> serve para separação da chave do valor
\n -> fim do comando

Register:

- “type | register ; username | adm ; password | adm” RMI -> MULTICAST
- “type | status ; operation | success” MULTICAST -> RMI
- “type | status ; operation | failed” MULTICAST -> RMI

Login:

- “type | register ; username | adm ; password | adm” RMI -> MULTICAST
- “type | status ; operation | success” MULTICAST -> RMI
- “type | status ; operation | failed” MULTICAST -> RMI

Search:

- “type | search ; text | oleole ; username | adm” RMI -> MULTICAST
- “type | search ; item_count | 1 ; value | asdasd” MULTICAST -> RMI

newURL.

- “type | newURL ; url | www.url.com” RMI -> MULTICAST
- “type | status ; operation | success” MULTICAST -> RMI
- “type | status ; operation | failed” MULTICAST -> RMI

Logout:

- “type | logout ; username | adm ; password | adm” RMI -> MULTICAST
- “type | status ; operation | success” MULTICAST -> RMI
- “type | status ; operation | failed” MULTICAST -> RMI

10MostImportat:

- “type | 10MostImportant” RMI -> MULTICAST
- “type | 10MostImportant ; value1 | www.xxas.pt ; value2…” MULTICAST -> RMI

10MostSearched:

- “type | 10MostSearched” RMI -> MULTICAST
- “type | 10MostSearched ; value1 | www.xxas.pt ; value2…” MULTICAST -> RMI

urlReferences:

- “type | urlReferences ; text | oleole ; username | adm” RMI -> MULTICAST
- “type | urlReferences ; item_count | 1 ; value | asdasd” MULTICAST -> RMI

get_notifications:

- “type | get_notifications ; text | oleole ; username | adm” RMI -> MULTICAST
- “type | get_notifications ; item_count | 1 ; value | asdasd” MULTICAST -> RMI

give_previlege:

- “type | give_privilege ; username | oleole” RMI -> MULTICAST
- “type | status ; operation | success” MULTICAST -> RMI
- “type | status ; operation | failed” MULTICAST -> RMI

getOnlineServer:

- “type | getOnlineServer” MULTICAST -> RMI
- “type | getOnlineServer ; id_server | 1 ; amount | 1” MULTICAST -> RMI

Distribuição de tarefas

A distribuição de tarefas utilizada foi uma das sugeridas no enunciado: *“Elemento 1 será responsável pelo Multicast Server e o elemento 2 pelo RMI Server e pelo RMI Client. Esta divisão assume que o protocolo multicast é especificado inicialmente e que as alterações serão daí em diante mínimas.”* sendo que a parte do multicast foi realizada pelo aluno Ricardo Martins e a parte do RMI pelo aluno Rodrigo Rafael.

Tabela de testes

Requisitos Funcionais		Pass/Fail
Registar novo utilizador	Registar admin	Pass
	Registar user normal	Pass
	Negar registo a user existente	Pass
Acesso protegido com password (exceto pesquisas)	É posivel fazer pesquisas sem registo	Pass
	Apenas admin tem acesso as funcionalidades de admin	Pass
	Apenas users registados têm histórico	Pass
	Apenas users registados podem pesquisar por url	Pass
Indexar novo URL introduzido por administrador		Pass
Indexar iterativamente ou recursivamente todos os URLs encontrados		Pass
Pesquisar páginas que contenham um conjunto de palavras	Apenas mostra as páginas com ambas as palavras	Pass
Resultados ordenados por número de ligações para cada página	Ordena as páginas por número de vezes referenciadas	Pass
Consultar lista de páginas com ligações para uma página específica		Pass
Consultar lista de pesquisas feitas pelo próprio utilizador	Apenas retorna as pesquisas de um utilizador	Pass
Dar privilégios de administrador a um utilizador	Apenas dá previlégios a utilizadores registados	Pass

	Apenas dá previlégios a utilizadores que não são admin	Pass
Página de administração atualizada em tempo real		Fail
Notificação imediata de privilégios de administrador (online users)		Fail
Entrega posterior de notificações (offline users)	Utilizadores recebem notificações pendentes quando se ligam	Pass
Tratamento de Exceções		
Avaria de um servidor RMI não tem efeito visível nos clientes	O sistema liga-se ao backup	Pass
	O cliente pode continuar a usar a aplicação com o backup	Pass
Servidor RMI secundário testa e substitui o primário em caso de avaria longa		Pass
Em caso de avaria longa os clientes RMI ligam ao secundário (sessão mantida)		Pass
Avarias temporárias (<30s) dos servidores multicast são invisíveis para clientes	Receber resposta após falha inferior a 30s	Pass
Pedidos são garantidamente processados por N>=1 servidores multicast	Desde que haja um multicast a funcionar os pedidos são processados	Pass
Pedidos de indexação são respondidos apenas por um servidor multicast	Apenas um servidor envia resposta	Pass
	Servidores online ignoram pedidos de indexação que não têm o seu id	Pass

Failover		
O serviço funciona se houver pelo menos um servidor multicast disponível	O servidor recebe pedidos com apenas um multicast	Pass
	O servidor envia respostas com apenas um multicast	Pass
	O servidor atualiza os dados permantetes	Pass
Os servidores multicast recuperam de disco o seu estado se avariarem	Se um multicast for abaixo quando recomeça vai ler a informação guardada	Pass
Cada servidor multicast replica a sua parte do índice por outros servidores (TCP)		Pass
Cada servidor distribui URLs para serem indexados por outros servidores		Pass
O servidor RMI original, quando recupera, torna-se secundário		Pass
Extra (até 5 pontos)		
Servidores multicast replicam o índice repartido em grupos (5p)	Cada servidor envia a lista de diferenças desde a ultima partilha	Pass
Balanceamento da carga dos servidores multicast (3p)		Pass